CS6640-Assignment 3

Atul Rungta
U0734304
atul.rungta@utah.edu

1. Affine Image Transformation

a. Individual transformations:

i. Translation:

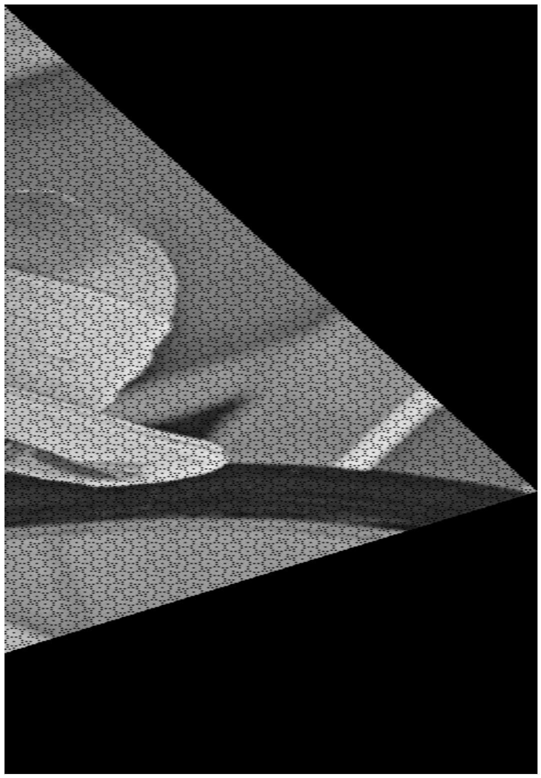
The translation was done after converting the image coordinates to homogenous coordinates: **[x, y, 1]**

Original Image

Translated Image

The translated image has portions of it clipped off because of the translation while the empty region is filled with black.

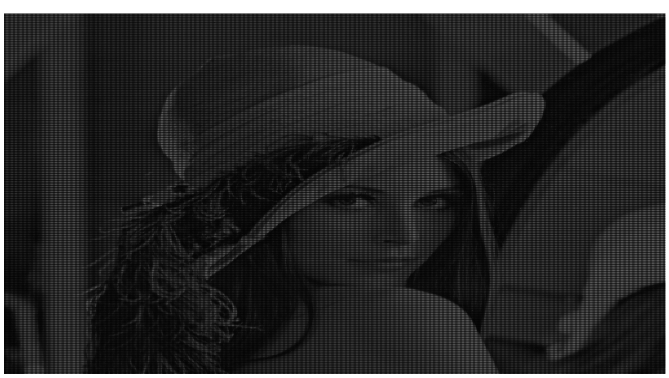
ii. Rotation:



Rotation

As expected, during the forward transform a lot of ‘holes’ develop in the target image due to missing values. Again, part of the image is clipped due to the rotation and the rest of the image is filled with black. The ‘holes’ were fixed when during the reverse transform.

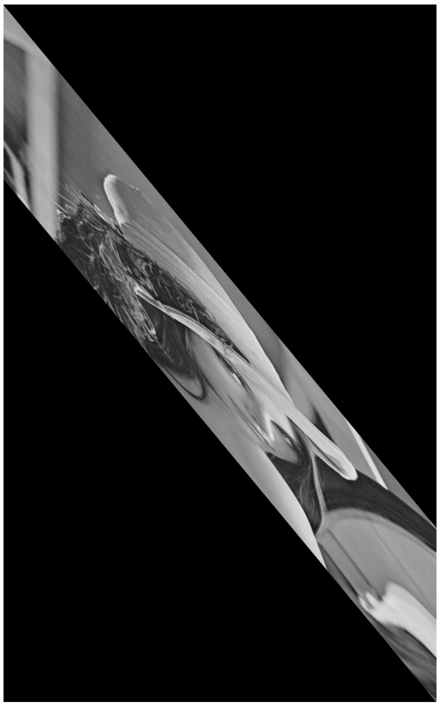
iii. Scaling:



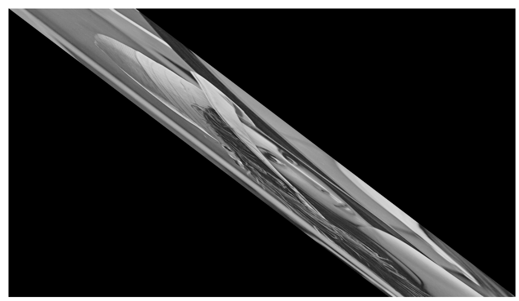
Scaling

The image was scaled by a factor of 2. Again, as expected there are holes in the target image because of missing values. Another interesting aspect of scaling is the decrease in intensity values because of the intensity getting spread across the larger image.

iv. Shearing:



Shear X



Shear Y

The shearing surprisingly shows no holes, or it could be because of the small size of the image.

b. Affine transform with Nearest Neighbor interpolation:

The transform was applied in the following order:

Translation->Rotation->Scaling->ShearX->ShearY

Affine transform of the Image

The nearest neighbor is really simple to implement but the combined affine transform makes the image barely recognizable. This makes it difficult to see the artifacts upon zooming. The backward rotation transformation makes it much easier to see the results properly:

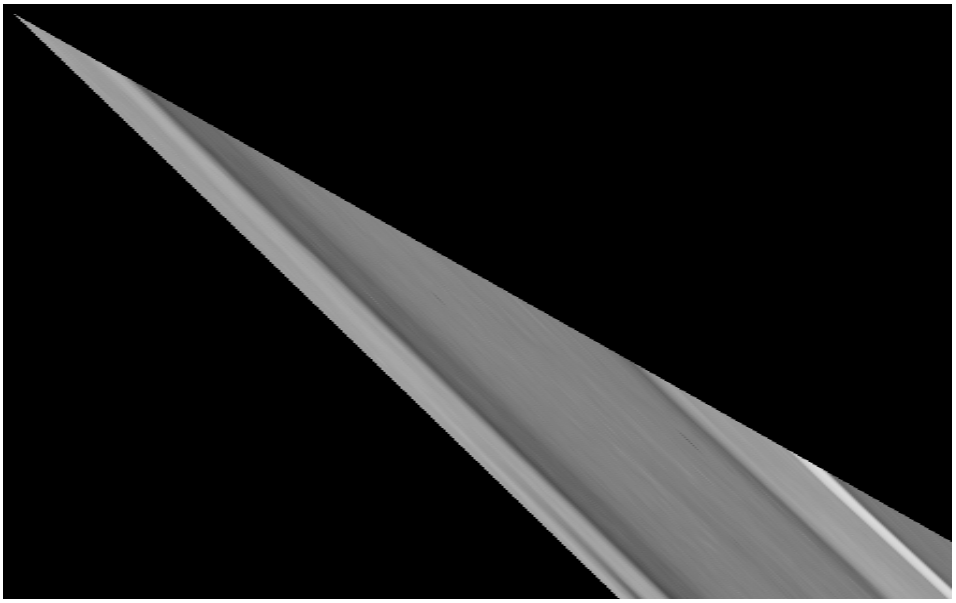
Rotation

Zoomed Image (Not done using Scaling)

As is apparent, the nearest neighbor interpolation, although simple, creates a lot of ‘jaggies’.

c. Affine transform with Bilinear interpolation:

The same order of transformation was used in case of bilinear interpolation:



Affine transform of the Image

As in case of nearest neighbor interpolation, the bilinear interpolation yields indiscernible output. To show the effects properly, I will show the interpolation on rotation of the image:

Rotation

Here is the zoomed in image of the same to show the Bilinear interpolation is better than nearest neighbor (The image was not zoomed using scaling):

Zoomed Image

2. Affine transform from Landmarks:

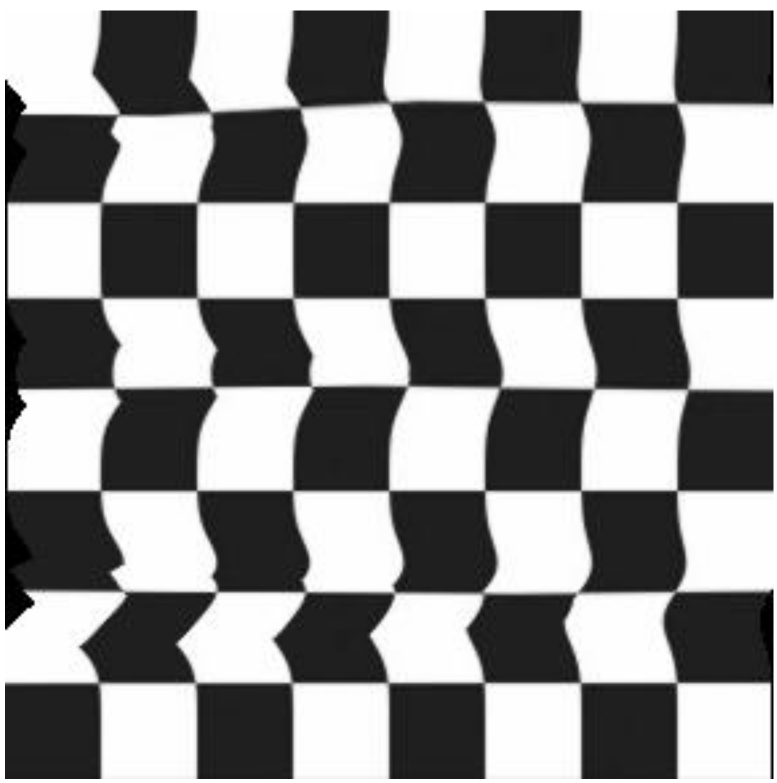
For this part of the assignment, I first found an image pair which was already transformed:

A total of 5 landmark points were used to reconstruct the transform. The pseudo-inverse was used to calculate the inverse transform. The calculated transform was again applied using bilinear interpolation:

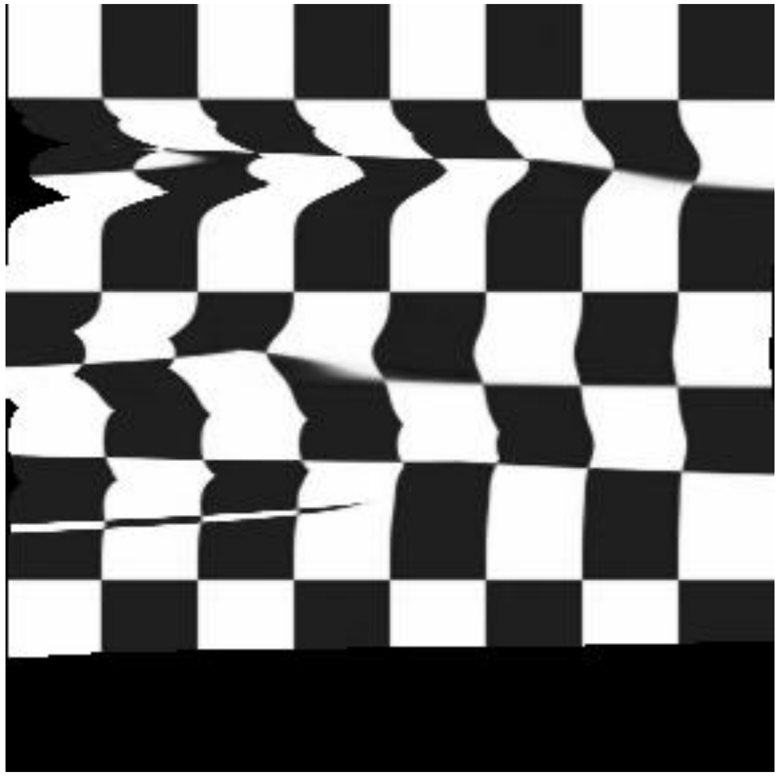

Retransformed Image

3. Image Warping using Radial Basis Functions:

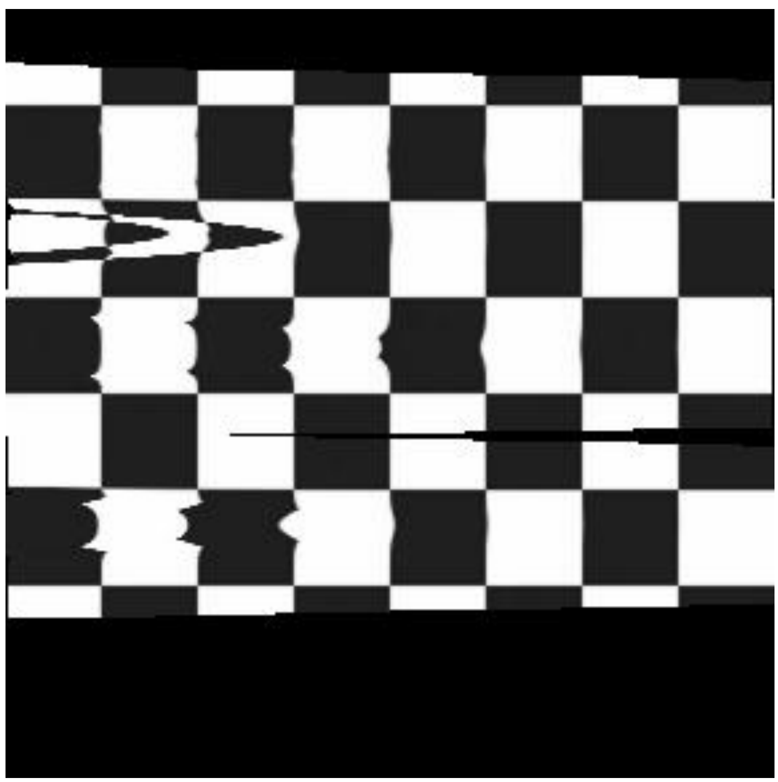
The radial basis functions were calculated as mentioned in the course notes using 5 landmark points and the checkerboard image as the source. Surprisingly, for values of sigma < 10 the results were disappointing. So, I used sigma = 10.5 for my code. Using the forward transform resulted in a lot of holes and tearing of the image as expected. So, I implemented the reverse as mentioned in the course notes. The result was encouraging:


Warped Image(sigma 10.5)

The landmark points were chosen at random for the 3 images.



Warped Image (sigma 7)



Warped Image(sigma 4)

The landmark points were chosen at random for the 3 images.